2010

ICMP Covert Channel Resiliency

Kristian Stokes
Rochester Institute of Technology

Bo Yuan
Rochester Institute of Technology

Daryl Johnson
Rochester Institute of Technology

Peter Lutz
Rochester Institute of Technology

ICMP Covert Channel Resiliency

Kristian Stokes, Bo Yuan, Daryl Johnson, and Peter Lutz

Department of Networking, Security, and Systems Administration
B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York 14623

Abstract. The ICMP protocol has been widely used and accepted as a covert channel. While the ICMP protocol is very simple to use, modern security approaches such as firewalls, deep-packet inspection and intrusion detection systems threaten the use of ICMP for a reliable means for a covert channel. This study explores the modern usefulness of ICMP with typical security measures in place. Existing ICMP covert channel solutions are examined for compliance with standard RFCs and resiliency with modern security approaches.

Key words: covert channels, ICMP protocol, firewall, ICMP-chat, ping tunnel, formation hiding

1 Introduction

The Internet Control Message Protocol (ICMP) [1] is designed to provide feedback about problems in the communication environment. ICMP relies on the basic support of IP as part of a higher level protocol. Due to this dependency, both ICMPv4 and ICMPv6 exist for both versions of IP. Many message codes exist within ICMP to properly diagnose network problems and traffic flow. ICMP messages are sent under many different circumstances such as an unreachable destination or general congestion control on the network. Simple network troubleshooting utilities such as ping and traceroute utilize explicit ICMP messages to gather information about a network.

Covert channels often refer to a hidden information stream and more specifically, hidden streams embedded in IEEE 802 networks. Lampson [2] originally defined covert channels under a number of categories such as storage, timing, termination, resource exhaustion and power. Most covert channels involving the use of ICMP are largely storage channels where unused fields are utilized for covert communication. ICMP as a covert channel provides many benefits due to the overall simplicity. Only several fields exist within most ICMP messages, which enable quick implementations and simple channel setup/teardown. The idea of using ICMP as a covert channel is to use a lesser standard communication protocol rather than TCP or UDP. This will have a smaller footprint across the network and may go unnoticed by network administrators and traffic analyzers. What really makes the ICMP protocol a viable covert channel is the use of data fields or payloads within certain messages. By generating packets based on specific message codes and embedding the actual covert channel message in the data field enables ICMP to serve as an alternate use for covert channels. These simple factors enable ICMP to be considered as stealth traffic.

2 Current Countermeasures

Countermeasures exist within the ICMP covert channel realm, however they come at a cost. As with all aspects of security, tradeoffs exist.

Blocking all ICMP traffic from entering the network prevents all aspects of ICMP communication, including covert channels. This methodology may not be acceptable due to the loss of network troubleshooting abilities. Also, blocking ICMP communications at a central firewall may not solve the problem due to the ability to send ICMP messages from an internal network to other internal hosts.

Block specific ICMP messages from entering the network. This methodology also incorporates the use of segmenting incoming and outgoing connections. If this countermeasure is enacted, both parties involved in the covert channel can develop fuzzing techniques to modify the ICMP messages used for communication or which party initiates the channel depending on the traffic restrictions.

Restricting the size of ICMP packets. By blocking large ICMP packets, an ICMP message with an extensive data field will be dropped and perceived as a

crafted packet with a malicious or unknown payload. Large ICMP packets can be used to test a network for proper handling of large packets. An adversary can also overcome this limitation by fragmenting their ICMP covert channel to smaller packets.

Traffic normalization. Traffic normalization techniques such as state preservation will reply and generate new packets on the senders behalf. The normalizer, typically a firewall, will serve as a proxy, rebuild messages and construct new payloads. This activity essentially disrupts ICMP covert channel communication by stripping out the message payload. State preservation requires significant computational power and may not scale to particular environments with high traffic loads.

3 Related Work

Research in this area focuses on existing solutions currently available for ICMP covert channel communication. Loki, an ICMP tunneling back door application, tunnels remote shell commands in ICMP echo reply / requests and DNS query / reply traffic. This proof of concept was originally published in Phrack World News [3] to demonstrate the vulnerabilities within these protocols. This implementation is very easy to deploy and thusly carries a risk security risk of ICMP tunneling.

Another implementation named Ping Tunnel [4], is focused on reliably tunneling TCP connections using ICMP echo request and reply packets. This tool can be with an outside proxy to tunnel traffic using ICMP messages back to the requesting client. The exterior proxy serves all TCP requests and forwards the data back to the client via ICMP echo reply. Although this solution can be viewed as subverting typical network communication, it is also important to realize other potential uses for ICMP covert communication.

ICMP-Chat [6] implements a basic console-based chat mechanism that utilizes ICMP packets for communication. A unique aspect of this solution incorporates the use of an encrypted data field using AES-256. By implementing an encrypted payload this further secures the covert channel, but adds increased suspicions on an abnormal ICMP payload.

With the increasing awareness of IPv6, covert channel tools further expanded into the realm of ICMPv6. A tool v00d00N3t [7] was developed to operate specifically over IPv6. This tool focuses on the infancy of IPv6 protection technology. Dual IPv4/IPv6 routers are utilized to route IPv6 traffic to send messages and files using ICMPv6. Useful information is embedded into fields other than the data field. The ICMPv6 ID is used to identify how many bytes out of the payload to read. The ICMPv6 sequence number tells the receiver if it should read the packet.

Additional defensive research has been performed to further limit the capabilities of ICMP covert channels. A Linux kernel module was developed to scan ICMP messages for specific signatures such as passwd, root, etc, ls and dir [5]. If these signatures were detected, the ICMP message was essentially scrubbed

by zeroing out the data field while being processed by the network stack. This technique is similar to normalizing traffic, however the action of data scrubbing is performed on the end nodes. This solution can also be deemed as a very computational intensive process involving bi-directional deep-packet inspection. The added network processing overhead may not be acceptable for performance reasons.

With these current solutions outlined, little information is available for actual survivability of the tools and general ICMP covert channel message resiliency with common security appliances. The use of intrusion detection, intrusion prevention and firewalls are commonplace within a production environment. To fully understand the capabilities of ICMP as a covert channel, it must contain a moderate amount of resiliency in modern networks.

4 Existing Solutions

ICMP-Chat [6] is based on a simple console-based chat interface which uses ICMP packets for communication. Features of this solution include the use of AES-256 encryption for encrypting chat data across the channel. ICMP-Chat provides several protection mechanisms such as password protecting the channel using SHA-256 and supporting the ability to change usage of ICMP codes within the application. This allows for mobility between ICMP codes to further obfuscate the channel.

Ping Tunnel [4] allows a user to tunnel TCP connections to a remote host using ICMP echo request and reply packets. This is achieved through the use of a proxy serving as the remote host. This solution can be utilized as a covert channel when the operating network is heavily restricted.

Each of these solutions will be tested and further explored in the following sections in terms of resiliency with modern security devices such as firewalls and intrusion detection systems.

5 Experiment

A minimalistic test environment was created using VMware Workstation. Two CentOS workstations are used as the endpoints of the covert channel. A Checkpoint NGX R65 firewall separates the network between internal and external clients. The internal network simulates a small business with a firewall and IDS in place. The external network simulates an Internet node communicating to the business. ICMP communication is permitted to the interior node, all remaining traffic is dropped at the firewall.

The Checkpoint firewall serves as a network layer inspection engine to detect fields and patterns in the network layer. This firewall provides sufficient inspection for a network layer covert channel.

A Snort IDS [8] is deployed inline on the network listening in promiscuous mode. All traffic passing through the internal network is captured and analyzed

by the IDS. The standard Snort ruleset was updated to the latest version from Sourcefire.

The basis for this experiment focuses on the resiliency of ICMP as a viable covert channel. ICMP-Chat and Ping Tunnel will be used in this environment to test resiliency against a modern firewall and IDS. Resiliency will be examined for each tool on connection establishment, connection integrity and overall covert design.

6 Results

Given the provided experiment, both ICMP-Chat and Ping Tunnel are examined for ICMP resiliency. Each tool was installed and tested for functionality on both internal and external workstations.

6.1 ICMP-Chat

By default, ICMP-Chat uses echo reply packets for primary communication. Figure 1 illustrates the network topology of the experiment. When comparing standard operating system echo reply packets to echo reply packets generated by the ICMP-Chat application many parameters follow standard RFC compliance.

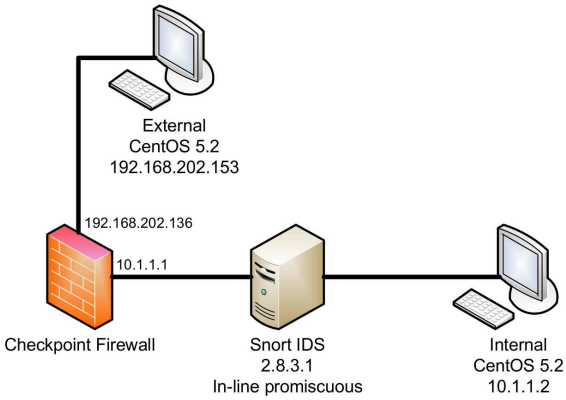

Fig. 1. ICMP-Chat covert channel environment

The standard CentOS echo reply, shown in Figure 2 consists of several key focus areas such as packet size and data content. A total of 98 bytes are captured on the wire with a 56 byte data field. The data field also standardizes on the following sequence of characters: !#$%&'()*+,-./01234567. For a perfect covert channel, the ICMP communication should match this similar format to reduce detection.

```
     Frame Frame 2397 (98 bytes on wire, 98 bytes captured)
     Ethernet II, Src: Vmware_3a:02:dd (00:50:56:3a:02:dd), Dst:
Vmware_3a:07:f8 (00:50:56:3a:07:f8)
     Internet Protocol, Src: 10.1.1.2 (10.1.1.2), Dst: 192.168.202.153
(192.168.202.153)
     Internet Control Message Protocol
  Type: 0 (Echo (ping) reply)
  Code: 0 ()
  Checksum: 0x2568 [correct]
  Identifier: 0x2448
  Sequence number: 1 (0x0001)
  Data (56 bytes)
     Data: CB9E0B4AEC62080008090A0B0C0D0E0F1011121314151617

     0000  00 50 56 3a 07 f8 00 50 56 3a 02 dd 08 00 45 00   .PV:PV:.E.
     0010  00 54 84 94 00 00 40 01 5f d0 0a 01 01 02 c0 a8   .T.@._.....
     0020  ca 99 00 00 25 68 24 48 00 01 cb 9e 0b 4a ec 62   .%h$H..J.b
     0030  08 00 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15   ............
     0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25   ..... !#$%
     0050  26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35   &()*+,-./012345
     0060  36 37                                            67
```

Fig. 2. Standard CentOS echo reply

An ICMP-Chat echo reply session was initiated, sent with the message test and the capture is illustrated in Figure 3.

```
     Frame 491 (318 bytes on wire, 318 bytes captured)
     Ethernet II, Src: Vmware_3a:02:db (00:50:56:3a:02:db), Dst:
Vmware_3a:07:f7 (00:50:56:3a:07:f7)
     Internet Protocol, Src: 192.168.202.153 (192.168.202.153), Dst: 10.1.1.2
(10.1.1.2)
     Internet Control Message Protocol
  Type: 0 (Echo (ping) reply)
  Code: 0 ()
  Checksum: 0x8857 [correct]
  Identifier: 0x0000
  Sequence number: 1280 (0x0500)
  Data (276 bytes)
     Data: 000000000000000000000000000000000000000000449569F8...

     0000  00 50 56 3a 07 f7 00 50 56 3a 02 db 08 00 45 00   .PV:...PV:....E.
     0010  01 30 eb f9 00 00 40 01 f7 8e c0 a8 ca 99 0a 01   .0....@.........
     0020  01 02 00 00 88 57 00 00 05 00 00 00 00 00 00 00   .....W..........
     0030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 44 95   ..............D.
     0040  69 f8 98 69 2a cf 87 6c 41 eb bf 95 14 0c 7e b8   i..i*..lA.....~.
     0050  af 0c d4 17 ad d0 2e 76 b8 75 cf 29 a4 8f 4c f7   .......v.u.)..L.
     0060  75 fe 22 86 a4 96 a6 14 0d 03 3c 2a 0d c5 95 9d   u."......<*....
     0070  ca cc 56 1d 39 f0 21 ab 1d a9 76 0c 32 ec 71 3a   ..V.9.!...v.2.q:
     0080  11 66 17 46 0a dc 5d ce 74 41 f8 bf e7 ee a1 1e   .f.F..].tA......
     0090  c2 b9 88 07 6a 18 f2 88 3c 3d 5d 29 a4 b9 49 ac   ....j...<=])..I.
     00a0  bd c3 91 ec 82 da 2b d9 eb 4d b7 8e 54 44 ef 04   ......+..M..TD..
     00b0  d4 91 4d 30 d7 91 c8 36 0f 92 d1 47 4b 50 e5 1f   ..M0...6...GKP..
     00c0  8e 68 c6 90 1d 9c a0 46 bb b0 2f 64 17 ed 60 ca   .h.....F../d..`.
     00d0  d9 81 89 b7 30 08 9d 80 c6 f6 cd 4e 7c 0f 82 67   ....0......N|..g
     00e0  15 04 fe e6 d9 8f 7b 20 a6 12 be 2c 90 0e b5 2a   ......{ ...,...*
     00f0  8d 16 56 60 d0 85 f8 60 1f 3d 54 25 d4 71 3b 95   ..V`...`.=T%.q;.
     0100  8e 24 82 68 b4 64 0d be 00 35 92 67 6e 4d 46 e0   .$.h.d...5.gnMF.
     0110  40 85 f2 a7 89 76 23 75 d4 72 7a 7e 94 4e 09 3e   @....v#u.rz~.N.>
     0120  5f 29 43 c7 ce ec 45 a1 f6 1a 76 7c 79 af 87 dc   _)C...E...v|y...
     0130  d1 63 26 fd 28 48 b3 63 ca 43 5f 82 94 00         .c&.(H.c.C_...
```

Fig. 3. ICMP-Chat echo reply

Major differences with this reply packet are centered on overall packet size of 318 bytes with a data size of 276 bytes. The dramatic increase in data size contributes to an abnormal attribute for typical ICMP traffic. This large packet size decreases the covertness of this solution. If the network restricts large ICMP packets, ICMP-Chat will likely be blocked.

Firewall Resiliency. Session initiation of ICMP-Chat with the Checkpoint firewall must follow request and reply structure. By default, ICMP-Chat uses echo reply packets. Similar to stateful firewall inspection, the Checkpoint firewall expects an echo request then permits an echo reply. If both internal and external nodes use the echo replies, the firewall does not permit the communication. If the communication is changed to an echo request and reply structure the Checkpoint firewall permits the communication. This similar structure must be followed for continued communication; an echo request must be received before an echo reply is permitted to traverse the firewall.

IDS Resiliency. The Snort IDS was unable to detect abnormal ICMP traffic when conducting the covert channel. This further confirms that abnormal ICMP packet sizes are not added to the normal IDS ruleset. Given that the data field is encrypted, this adds to the level of complexity needed for IDS detection in covert channels.

6.2 Ping Tunnel

Ping Tunnel [4] serves as a covert tunneling tool to disguise TCP traffic in ICMP request and reply packets. The basis of this technique is to disguise traffic as a wrapper protocol and bypass specific TCP filtering. This is achieved through the use of an external proxy to convert transmitted ICMP client packets back to standard TCP packets. Unlike ICMP-Chat, Ping Tunnel strictly uses ICMP echo request and reply packets for communication. Connections are very similar to TCP in that lost packets are resent as necessary to allow for reliability. Multiple connections are permitted through the use of the ICMP identifier field. The identifier field is included within the standard Ping Tunnel packet format and should not be confused with the ICMP sequence number field.

The example test environment, illustrated in Figure 4, was designed to allow the client node to establish an SSH tunnel to an Internet based server. A known proxy address was provided to the client and listens on a local port which tunnels all traffic via ICMP echo request/reply packets. Once the ICMP proxy was established, an SSH connection was initiated from the client to the localhost port.

Upon establishment of the connection, the standard SSH handshake can be viewed in captured ICMP echo request/reply packets.

The packet capture in Figure 5 shows a standard OpenSSH version handshake embedded in the data field of a ICMP echo reply.

Firewall Resiliency. The Checkpoint firewall permitted Ping Tunnel traffic largely due to the adherence to the ICMP echo request and reply structure. If this request and reply format is continued throughout the communication, the traffic will go largely unnoticed. Unsolicited connections will be dropped by the firewall. Similar to ICMP-Chat, the Checkpoint firewall did not specifically block large ICMP packets.

IDS Resiliency. The inline Snort IDS also failed to detect this covert channel. Again, abnormally sized ICMP messages are overlooked given this situation.

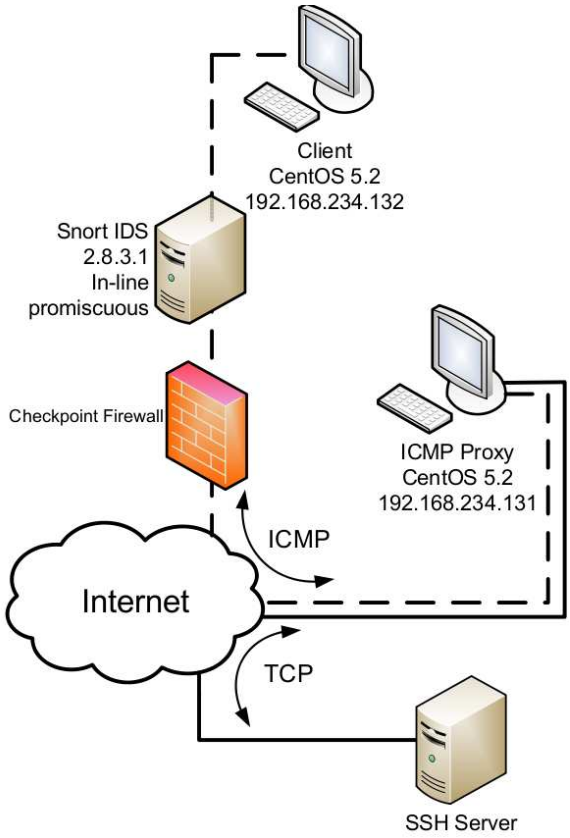

Fig. 4. Ping Tunnel covert channel environment

```
      No.     Time        Source                Destination
Protocol Info
    10 0.012010    192.168.234.131       192.168.234.132
ICMP     Echo (ping) reply

      Frame 10 (92 bytes on wire, 92 bytes captured)
      Ethernet II, Src: Vmware_3a:02:dd (00:50:56:3a:02:dd), Dst:
Vmware_3a:02:db (00:50:56:3a:02:db)
      Internet Protocol, Src: 192.168.234.131 (192.168.234.131), Dst:
192.168.234.132 (192.168.234.132)
      Internet Control Message Protocol
  Type: 0 (Echo (ping) reply)
  Code: 0 ()
  Checksum: 0x195d [correct]
  Identifier: 0xe510
  Sequence number: 0 (0x0000)
  Data (50 bytes)

      0000  d5 20 08 80 00 00 00 00 00 00 00 00 80 00 00 02   .
..............
      0010  00 00 00 00 00 00 00 15 00 00 e5 10 53 53 48 2d
............SSH-
      0020  32 2e 30 2d 4f 70 65 6e 53 53 48 5f 35 2e 31 0d
2.0-OpenSSH_5.1.
      0030  0a 20                                            .
      Data: D5200880000000000000000800000020000000000000015...
```

Fig. 5. Ping Tunnel OpenSSH handshake

It is possible to tailor a custom ruleset to detect this activity, but the default ruleset fails to recognize frequent, abnormally large ICMP messages.

7 Conclusion

Based on the experimental findings of this study it is very clear that a simple ICMP based covert channel can easily subvert many modern security appliances if general ICMP traffic is permitted. Administrators and security researchers should be aware of the capabilities of a seemingly helpful protocol. Current tools are widely and freely available for use across a number of platforms.

Blocking all ICMP traffic may not be an acceptable business practice in may cases. Steps can be taken to further reduce the risk of an ICMP covert channel. Limiting the overall packet size of ICMP messages may disrupt communications. Ensuring unsolicited ICMP messages are dropped at the perimeter can assist in preventing channel establishment, but as investigated, this can be easily circumvented by following a request and reply structure.

An IDS can be further improved by monitoring ICMP traffic flow. Ping Tunnel generated abnormally sized ICMP packets and often produced constant or bursting traffic to the proxy node. IDS technology can be implemented to flag this abnormal traffic for further investigation. The traffic analysis can be compared to constant streams of ICMP traffic resulting in varying packet sizes to a single destination.

Even with many security mechanisms and precautions in place, covert channel may still exist. This activity is essentially what a covert channel fundamentally strives to be, an undetectable channel of communication.

References

1. Postel, J. “INTERNET CONTROL MESSAGE PROTOCOL.” RFC 792. Internet Draft Submission Tool. 30 Apr. 2009.
2. Lampson, B. W. 1973. “A note on the confinement problem.” Commun. ACM 16, 10 (Oct. 1973), 613-615.
3. Daemon9. 1997. “LOKI2.” Phrack Magazine, Vol. 7 (51) http://www.phrack.com/issues.html?issue=51&id=1.
4. Stodle, Daniel. “Ping Tunnel.” http://www.cs.uit.no/ daniels/PingTunnel.
5. Singh, Abhishek, et al. 2003. “Malicious ICMP Tunneling: Defense against the Vulnerability.” In: Lecture Notes in Computer Science, Vol. 2727, Springer Berlin / Heidelberg, pp. 226-236.
6. Muench, Martin J. 2003. “ICMP-Chat.” http://icmpchat.sourceforge.net/index.html.
7. Murphy, R. P. 2006. “IPv6 / ICMPv6 Covert Channels.” DEFCON 14. https://forum.defcon.org/archive/index.php/t-7588.html.
8. Sourcefire. “Snort IDS.” http://www.sourcefire.com.